

MULTIMEDIA



UNIVERSITY

STUDENT ID NO

--	--	--	--	--	--	--	--	--	--

MULTIMEDIA UNIVERSITY

FINAL EXAMINATION

TRIMESTER 1, 2017/2018

ECP4136 – JAVA TECHNOLOGY
(RE / TE)

16 OCTOBER 2017
9.00 a.m. – 11.00 a.m.
(2 Hours) - OPEN BOOK

INSTRUCTIONS TO STUDENT

1. This examination paper consists of FIVE (5) pages including cover with THREE (3) questions only.
2. Answer ALL questions. The distribution of the marks for each question is given.
3. This is an Open-Book examination.
4. Please write all your answers in the Answer Booklet provided.

Instructions:

Figure 1 shows the UML diagram of **Temperature** interface, **Kelvin** class, **Celsius** class, **Fahrenheit** class, **Reaumur** class, **Rankine** class, and **TempConverter** class.

You are to "implement" the UML diagram in **Question 1** and "apply" the UML diagram in **Question 2** and **Question 3**.

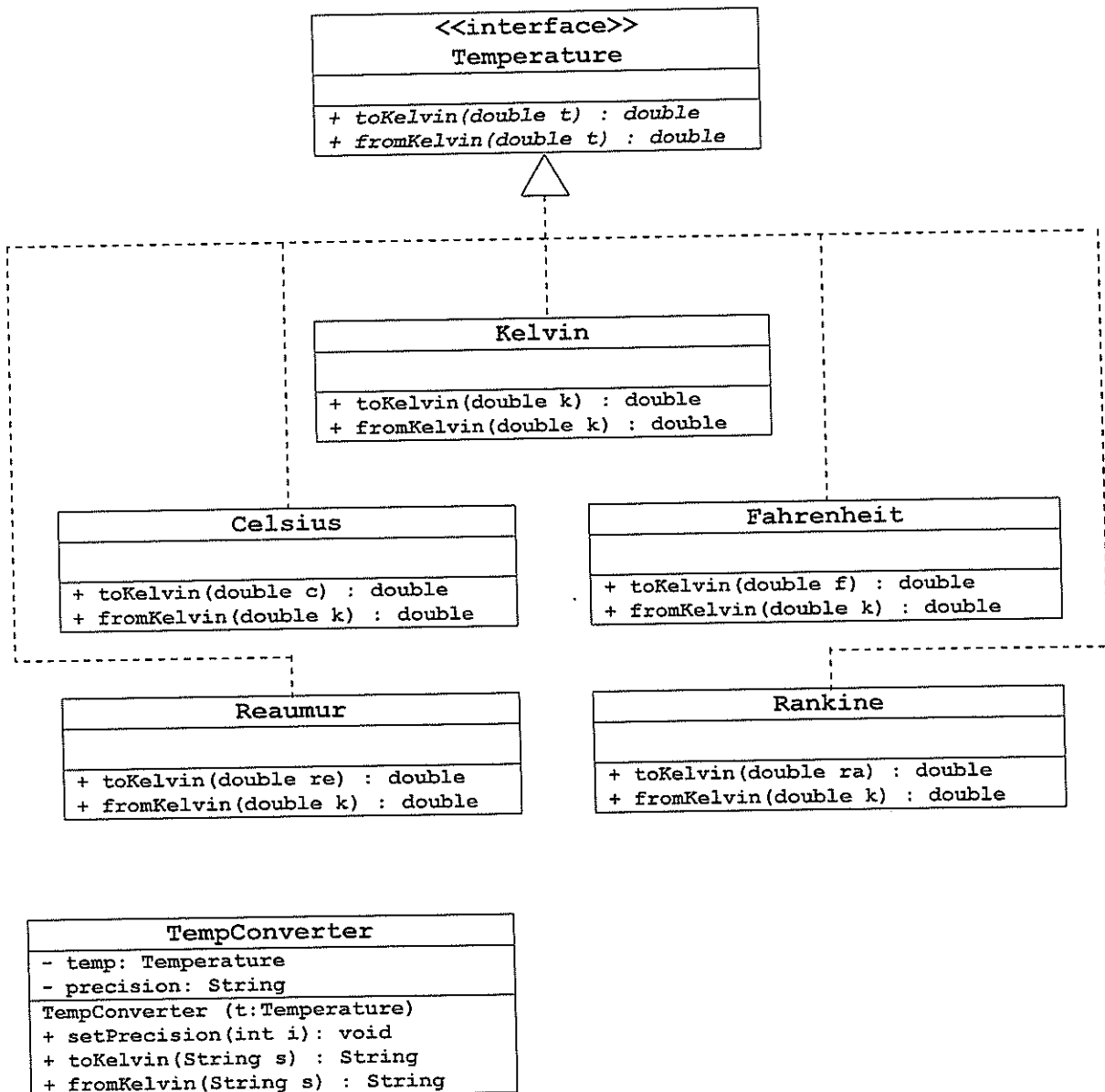


Figure 1

Continued...

Question 1

Implement the UML diagram shown in Figure 1. The marks distribution is as follows:

Creation of **Temperature** interface and its corresponding methods. [2 marks]

Implementation of the **Kelvin**, **Celsius**, **Fahrenheit**, **Reaumur**, and **Rankine** class with the following requirements:

	To Kelvin	From Kelvin	
Kelvin, k	k	k	[3 marks]
Celsius, c	$c+273.15$	$k-273.15$	[5 marks]
Fahrenheit, f	$5/9 (f+459.67)$	$9/5k -459.67$	[5 marks]
Reaumur, re	$5/4re +273.15$	$4/5(k-273.15)$	[5 marks]
Rankine, ra	$5/9ra$	$9k/5$	[5 marks]

Implementation of the **TempConverter** class with the following requirements:

- Declaration of **temp** variable and creation of a constructor that save a copy of the argument. [2 mark]
- Declaration and initialization of **precision** variable with the default value of `"%.2f"`. Implementing the **setPrecision** method based on the value **i**. For example, if **i** is 1, **precision** will be `"%.1f"`. [3 mark]
- Implementation of **fromKelvin** method whereby a string value of a temperature will be converted from Kelvin values. For example, calling **fromKelvin** ("100") will return "-173.15" if **temp** is a **Celsius** object and **precision** is `"%.2f"`. [5 mark]
- Implementation of **toKelvin** method whereby a string value of a temperature will be converted to Kelvin values. For example, calling **toKelvin** ("100") will return "55.6" if **temp** is a **Rankine** object and **precision** is `"%.1f"`. [5 mark]

Continued...

Question 2

Create a **TempTest** class that will create a console program to convert temperature from one unit to another. Figure 2 shows a sample output where the user is trying to convert 55 degree Celsius to its equivalent in Fahrenheit. The class has the following specifications:

Declare and initialize a **Scanner** object. [1 marks]

Create a static **+printAndGetChoice(title:String):int** method. The method will print a menu and ask the user to enter the choice, which will be returned. The menu consist of printing the title, for example "Convert from:", followed by the 5 different temperature unit. [9 marks]

Create a static **+getTemperature(choice:int):Temperature** method. The method will return the appropriate **Temperature** object based on the choice. [7 marks]

The **main** method has marks distribution is as follows:

- Declaration of **Temperature** object and a **choice** integer. [1 mark]
- Correct use of loop [1 mark]
- Printing menu(to), get choice, and get temperature object or exit [3 marks]
- Requesting for current value and converting it to Kelvin equivalent. [3 marks]
- Printing menu(from), get choice, and get temperature object or exit [3 marks]
- Displaying the equivalent value. [2 marks]

```
Convert from:
1. Kelvin
2. Celsius
3. Fahrenheit
4. Rankine
5. Reaumur
Enter 1-5 to continue, 0 to exit: 2
Enter current value: 55
Convert to:
1. Kelvin
2. Celsius
3. Fahrenheit
4. Rankine
5. Reaumur
Enter 1-5 to continue, 0 to exit: 3
Equivalent value is : 131.00
Convert from:
1. Kelvin
2. Celsius
3. Fahrenheit
4. Rankine
5. Reaumur
Enter 1-5 to continue, 0 to exit: 0
```

Figure 2

Continued...

Question 3

Create a `TempConverterPanel` class with a GUI output as shown in **Figure 3**. The specification of the class is as follows:

It has the following instance variables: [6 marks]

- `-toCelsiusRB: JRadioButton` with text "To Celsius" and default preselected
- `-toFahrenheitRB: JRadioButton` with text "To Fahrenheit"
- `-celsiusL: JLabel` with text "Celsius:"
- `-fahrenheitL: JLabel` with text "Fahrenheit:"
- `-celsiusTF: JTextField` with size of 10 and shows value 0.00 and
- `-fahrenheitTF: JTextField` with size of 10 and shows value "32.00"
- `-convertB: JButton` with text "Convert"

A constructor that perform the following actions:

- Configuring the button group, adding listener to all buttons, and setting the `celsiusTF` as un-editable. [6 marks]
- Adding all GUI components and setting the preferred size (230 x 130). [4 marks]

Creating a listener for the radio buttons where when the `toFahrenheitRB` is selected, `fahrenheitTF` is disabled and `celsiusTF` is enabled. Do the opposite if `toCelsiusRB` is selected. For example Figure 3a shows `toCelsiusRB` is selected and Figure 3b shows `toFahrenheitRB` is selected. [6 marks]

Creating a listener for the convert buttons where it will extract value from one of the text field, calculate the equivalent value and display it on the other text field. For example in Figure 3a, the user select `toCelsiusRB`, enter 100 in `fahrenheitTF`, click on the convert button. Thus the value 37.78 is shown in `celsiusTF`. In Figure 3b, the user would like to convert 50 degree Celsius to its Fahrenheit equivalent. Thus the value 122.00 is shown. [8 marks]

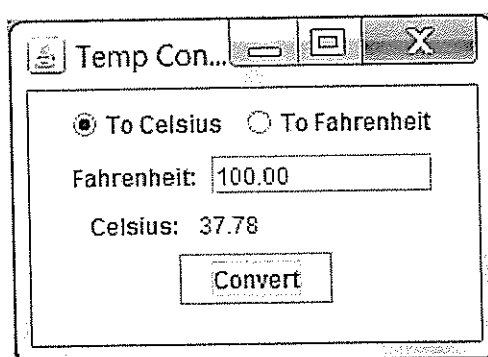


Figure 3a

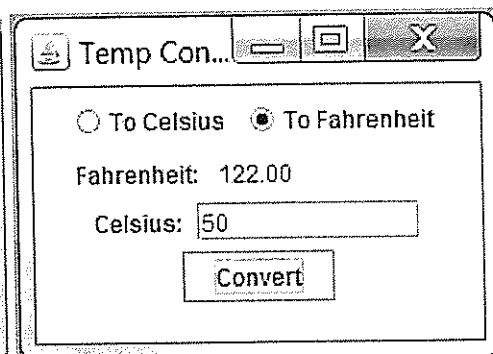


Figure 3b

End of Paper